PATENT

IN THE UNITED STATES PATENT & TRADEMARK OFFICE

Applicant	:	**Peter Brittingham, et al.**			
Application Number	:	**09/654,949**	**Confirmation No.**	:	**7078**
Filed	:	**September 1, 2000**	**Tech Cntr/AU**	:	**3714**
Attorney Reference	:	**128534-01801 (07028591)**	**Customer Number**	:	**26565**
Examiner	:	**Ross A. Williams**			
Title	:	**COMPUTER BASED TEST ITEM GENERATION**			

MAIL STOP APPEAL BRIEF
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

CERTIFICATE OF TRANSMISSION BY EFS
I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being transmitted by EFS to the United States Patent and Trademark Office, on the date shown below.

Dated: January 9, 2009

Signature: _____ (Sheena Hicks)

AMENDED APPEAL BRIEF

Sir:

This brief is submitted under 35 U.S.C. § 134 and is in accordance with 37 C.F.R. Parts 1, 5, 10, 11, and 41, effective September 13, 2004 and published at 60 Fed. Reg. 155 (August 2004). This brief is further to Appellant's Notice of Appeal, and in response to the Notification of Non-Compliant Appeal Brief dated January 8, 2009.

Table of Contents:

(1) Real Party in Interest:

The real party in interest is assignee Educational Testing Service.

(2) Related Appeals and Interferences:

No other appeals or interferences exist which relate to the present application or appeal.

(3) Status of Claims:

Claims 1-5, 13-17, and 19-20 are pending, finally rejected, and appealed herein.

(4) Status of Amendments:

No amendments are outstanding.

(5) Summary of Claimed Subject Matter:

As an initial matter, it is noted that according to the Patent Office, the concise explanations under this section are for Board convenience, and do not supersede what the claims actually state, 69 Fed. Reg. 155 (August 2004), see page 49976. Accordingly, nothing in this Section should be construed as an estoppel that limits the actual claim language.

As described in the abstract, for example, the invention generally pertains to creating test questions or "items" by generating variants from a test item model. More specifically, the invention identifies test item elements to be converted to variables ("variabilized"), indicates the range of values those variables can assume, and then generates test item variants with a simultaneous constraint solver. Generated test items can be stored and forwarded for use in a test. Test item models (which may include initial test items, see specification page 15 lines 6-8, FIG. 7 "stem" box) can be also stored for later use in further test item generation.

Claim 1 teaches a computerized method for creating test item models and generating test item variants, including obtaining a test item (page 15 line 6, FIG. 7 "stem" box), creating a test item model by identifying elements of the test item to be variabilized (page 17 lines 19-23, FIG. 7 "stem" box), variabilizing the elements by creating and defining the variables (page 15 line 18 to page 16 line 2, FIGs. 8 and 9 "stem" box), and

then generating a test item variant of the test item (page 29 line 21 to page 31 line 3, FIGs. 45-51) by assigning values to the variables using a simultaneous constraint solver, where the simultaneous constraint solver resolves a plurality of constraints pertaining to the variables (page 41 lines 1-3, FIGs. 14-15 and 21-23).

Claim 5 teaches a computerized method for generating test item variants, including identifying elements of a test item or a test item model to be variabilized (page 17 lines 19-23, FIGs. 8 and 9 "stem" box), variabilizing the identified elements by defining the variables (page 15 line 18 to page 16 line 2, FIGs 8 and 9 "stem" box) and specifying constraints for the variables (page 3 lines 16-17, page 24 line 20 to page 27 line 12, FIGs. 14-15 and 21-23), then using a simultaneous constraint solver to determine values for the variables based on the constraints (page 41 lines 1-3), and finally generating a test item variant with the determined values (page 3 lines 18-20, page 29 line 21 to page 31 line 3, FIGs. 45-51).

Claim 13 teaches a computerized method for generating test item variants from test item models, including retrievably storing test item models (page 4 lines 2-3, FIG. 4), selecting a test item model (page 37 lines 10-14, FIG. 74), simultaneously solving test item model constraints pertaining to variables of the selected test item model (page 3 lines 18-19, page 24 line 20 to page 27 line 12, FIGs 14-15 and 21-23), then generating test item solutions based on the selected test item model (page 41 lines 1-3, FIG. 77), and finally displaying, accepting and retrievably storing valid test item solutions (page 3 line 22 to page 4 line 2, FIG. 77).

(6) Grounds of Rejection to be Reviewed on Appeal:

(a) ***Whether claims 1-5, 13-16, and 19-20 have been properly rejected under 35 U.S.C. § 103(a) as unpatentable over Sweitzer and Swanson.***

(b) ***Whether claim 17 has been properly rejected under 35 U.S.C. § 103(a) as unpatentable over Sweitzer and Swanson.***

(7) Argument:

(a) *Claims 1-5, 13-16, and 19-20:*

It is noted that according to the Patent Office, a new ground of rejection in an Examiner's answer should be "rare" and should be levied only in response to such things as newly presented arguments by an Applicant or to address a claim that the Examiner previously failed to address, 69 Fed. Reg. 155 (August 2004), see, e.g., pages 49963 and 49980. Furthermore, a new ground of rejection must be approved by the Technology Center Director or designee and in any case must come accompanied with the initials of the conferees of the appeal conference, id., page 49979. Appellants note that the SPE signed off on the final rejections. Accordingly it is not expected that reopening of prosecution will occur, since the SPE has already had the chance to consider the gravamen of the arguments below and has rejected them.

Claims 1-5, 13-17, and 19-20 are rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 6,018,617 (*Sweitzer*) and U.S. Patent No. 5,657,256 (*Swanson*). Appellants assert that the cited prior art fails to teach or suggest numerous features claimed in the independent claims (1, 5, and 13) as required for a valid obviousness rejection, and that the rejections of these claims and their dependent claims are therefore improper and should be reversed. Claims 2-4 (which depend on claim 1) and claims 14-17 and 19-20 (which depend on claim 13) are patentable for at least the same reasons as their parent claims, described below. Dependent claim 17 is addressed separately. These dependent claims are further patentable based on the additional limitations that are added by each such dependent claim.

Appellants submit that *Sweitzer* and *Swanson* fail to disclose or fairly suggest, among other things, "generating a test item variant of the test item by assigning values to the variables using a simultaneous constraint solver" as recited in claim 1. Similarly, the cited prior art fails to teach or suggest "using a simultaneous constraint solver to determine values for the variables based on the constraints" as recited in claim 5, and "simultaneously solving test item model constraints pertaining to variables of the selected test item model" as recited in claim 13.

***Sweitzer* is directed to a method and a system for producing tests that includes the capability to format mathematical expressions. An authoring tool uses variation rules,**

which include an ordered list of definitions and constraints, to define instances of generalized problems. "To produce an instance of a problem, the list of variation rules is evaluated sequentially from top to bottom. If a constraint is not satisfied, the current pass through the list is abandoned and evaluation restarts from the top of the list. A valid instance of the problem results when the end of the variation rule is reached." *Sweitzer* at column 12 lines 41-46. In other words, *Sweitzer* uses a sequential constraint solver that "processes the variation rules for a problem from the top down." *Id.* at column 17 lines 49-50.

Appellants had previously amended claim 1 during prosecution to clarify that more than one constraint is evaluated simultaneously by embodiments of the present invention. Thus, in contrast to the sequential constraint solver of *Sweitzer*, claim 1 requires the use of a simultaneous constraint solver to resolve a plurality of constraints in order to generate test item variants. A simultaneous constraint solver solves for all constraints simultaneously. In other words, a simultaneous constraint solver may require determining values for a set of constraints only once to generate a test item variant. As a result, test items are generated more efficiently. In contrast, the sequential constraint solver of *Sweitzer* deletes computed constraint values and restarts from the beginning of its constraint sequence when a constraint is not satisfied. See *Sweitzer* at column 15 lines 39-40. As such, *Sweitzer*, unlike the simultaneous constraint solver of claims 1, 5, and 13, typically requires evaluating multiple constraints a plurality of times until all of the constraints in the sequence are satisfied. Accordingly, *Sweitzer* does not disclose "generating a test item variant of the test item by assigning values to the variables using a simultaneous constraint solver" as required by claim 1. Claims 5 and 13 are similarly distinct.

The Examiner asserts that *Swanson* cures the shortcomings of *Sweitzer* by disclosing a test item system that solves for multiple constraints (column 8 lines 5-47). *Swanson* discloses only that an important quality of a test construction model is "coming close as possible to meeting all constraints simultaneously" (column 8 lines 5-15). While this may be an admirable goal, Appellants respectfully disagree with any assertion that the combination of *Sweitzer* and *Swanson* provide any implemented solution that includes a simultaneous constraint solver as provided by embodiments of the present invention.

Indeed, *Swanson* actually teaches away from the assertion that it would be obvious to one of ordinary skill to modify *Sweitzer* to "use a constraint solver to satisfy or simultaneously solve for multiple constraints". See for example column 6 line 65 to column 7 line 5 of *Sweitzer*, which are statements that note the infeasibility of a prior art approach to a solution to the recognized problem, as the Examiner notes in the most recent office action. However, neither *Swietzer* nor *Swanson* then proceed beyond the decried prior art limitations regarding simultaneous constraint solution, except to note that the problem is "difficult or impossible to satisfy" with binary programming models (*Swanson* column 8 lines 5-10 and column 6 line 65 to column 1 line 1, as cited), and that a less than complete solution that softens constraints to mere "desired properties" (*Swanson* column 8 line 12) may be the best they can achieve.

(b) *Claim 17:*

Dependent claim 17 warrants particular discussion and separate appellate review. Appellants respectfully disagree with the Examiner's assertion that it would have been "an obvious design choice to a person of ordinary skill in the art to use C++ and variation rules language to simultaneously solve test item model constraints" and that using PROLOG IV and Test Creation Assistant (TCA) constraint language is not for a particular purpose nor solving a stated problem.

The specification provides that the present invention's preferred embodiments use PROLOG as its simultaneous constraint solver (page 41 lines 1-2). The specification states on page 45 lines 5-10 that the "TCA constraint solver can solve linear constraints and a large class of nonlinear constraints" and "returns all the solutions to the specified constraints", i.e. the invention is for a particular purpose and is solving the stated problem. On page 62 line 22, the specification clearly notes that "Constraints are solved all at once as a whole." This point is made yet again from page 66 line 19 to page 67 line 4, quoted here:

> ***"8. Constraints are Solved as a Whole.***
>
> ***All the constraints specified for one constraint are all solved as a whole, and not partially. This is particularly important in the case of the TCA where constraints are entered on different lines without any explicit operators (e.g. comma or semicolon) combining them (TCA supplies the default comma-operator (i.e.***

conjunct) between adjacent constraints) and thus one might get the incorrect impression that the constraints are solved independently."

In particular, page 41 lines 15-18 of the specification denote that a specified expression scanner is generated that "breaks the mathematical constraints into individual words and operators (called tokens) which are further used by the Prolog-expression parser." Page 42 lines 2-8 describe that the parser "recognizes the syntactic patterns of mathematical constraints" and "transforms the mathematical constraints into appropriate Prolog clauses, and calls Prolog IV to solve those clauses." An API is also provided to allow other programs to "access the constraints-solver to solve mathematical constraints." (page 42 lines 10-14). There is no suggestion of these processes in the prior art.

Contrasting the embodiments of the present invention with conventional programming, on page 43 line 16 to page 44 line 1, the specification details that the "TCA constraint language differs from procedural languages (e.g. C, FORTRAN) principally in that it is a goal-oriented language, that is users need specify only the constraints to be solved, and not how to solve them." C++ is a procedural language. Further, in TCA, program flow "is controlled by the constraint solver" and is "constraint-order independent". In contrast, page 63 lines 1-6 explain that unlike TCA, procedural languages have to provide a large number of program-flow control mechanisms. In lines 8-11 of page 63, the specification explains that because "the TCA constraint language uses relations to implement most functions and operators, one can use the same function [operator] to map a set of arguments to a result, or to map a result back to a set of arguments. … In procedural languages, one has to explicitly apply the reverse function to achieve the effect illustrated above." Also, as noted in lines 16-17, "TCA constraint language has no value-storage and no assignment." Given these fundamental differences, writing and solving constraints in TCA is not an variation of prior art attempts that would be obvious to one of ordinary skill in the art.

In summary, as the various embodiments of the present invention include features neither taught nor suggested by the cited prior art references, either separately or together, the obviousness rejections cannot be sustained. Mere untaught possibilities are insufficient to defeat patentability, and a prior art reference must be considered in its entirety, i.e. as a

whole. W.L. Gore & Associates, Inc. v. Garlock, Inc., 721 F.2d 1540, 220 USPQ 303 (Fed. Cir. 1983), cert. denied, 469 U.S. 851 (1984). Therefore, reversal is respectfully requested.

Conclusion

For the reasons advanced above, Appellants assert the rejected claims are indeed patentable, thus the rejections should be reversed.

Respectfully submitted,

MAYER BROWN LLP

By: Marc D. McSwain

Marc D. McSwain
Registration No. 44,929
Direct No. (650) 331-2048

Date: January 9, 2009

Customer Number 26565
MAYER BROWN LLP
P.O. Box 2828
Chicago, IL 60690-2828

APPENDIX A – APPEALED CLAIMS

1. A computerized method for creating test item models and generating test item variants comprising:

obtaining a test item;

creating a test item model by:

identifying elements of the test item to be variabilized,

variabilizing the elements to create variables, and

defining the variables; and

generating a test item variant of the test item by assigning values to the variables using a simultaneous constraint solver, wherein the simultaneous constraint solver resolves a plurality of constraints pertaining to the variables.

2. The method according to claim 1, wherein said model creation further comprises specifying constraints that define a relationship among the variables.

3. The method according to claim 2 further comprising accepting and retrievably storing the test item variant.

4. The method according to claim 3 further comprising accepting and retrievably storing the test item model.

5. A computerized method for generating test item variants, the method comprising:

identifying elements of a test item or a test item model to be variabilized;

variabilizing the identified elements;

defining the variables;

specifying constraints for the variables;

using a simultaneous constraint solver to determine values for the variables based on the constraints; and

generating a test item variant with the determined values.

13. A computerized method for generating test item variants from test item models comprising:

retrievably storing test item models;

selecting a test item model;

simultaneously solving test item model constraints pertaining to variables of the selected test item model and generating test item solutions based on the selected test item model; and

displaying, accepting and retrievably storing valid test item solutions.

14. The computerized method of claim 13 wherein retrievably storing test item models comprises:

obtaining a test item;

identifying elements of the test item to be variabilized;

variabilizing the elements to create the variables;

defining the variables; and

accepting the variabilized test item with defined variables as a test item model.

15. The computerized method of claim 13 further comprising specifying constraints that define the relationship among the variables.

16. The computerized method of claim 14 further comprising displaying and retrievably storing the accepted test item model.

17. The computerized method of claim 14 wherein the test item model constraints are simultaneously solved using PROLOG IV and Test Creation Assistant constraint language.

19. The computerized method of claim 14 wherein variables can be defined by values which are variables.

20. The computerized method of claim 15 wherein the variables are new variables for which new constraints are defined as needed.

APPENDIX B – EVIDENCE

None (this sheet made necessary by 69 Fed. Reg. 155 (August 2004), page 49978).

APPENDIX C – RELATED PROCEEDINGS

None (this sheet made necessary by 69 Fed. Reg. 155 (August 2004), page 49978).